H 5

```
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFFFFFFFFFFFFFF         111              111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                  111111           111111        XXX         XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFFFFFFF.FFF            111              111              XXX
FFFFFFFFFFFF            111              111              XXX
FFFFFFFFFFFF            111              111              XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111           XXX   XXX
FFF                     111              111        XXX         XXX
FFF                     111              111        XXX         XXX
FFF                     111              111        XXX         XXX
FFF                  111111111        111111111     XXX         XXX
FFF                  111111111        111111111     XXX         XXX
FFF                  111111111        111111111     XXX         XXX
```

```
_$25
Symb
----
IOC$
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

FILEID**CHKPRO

```
  CCCCCCCC  HH      HH  KK      KK  PPPPPPPP    RRRRRRRR      000000
  CCCCCCCC  HH      HH  KK      KK  PPPPPPPP    RRRRRRRR      000000
CC          HH      HH  KK      KK  PP      PP  RR      RR  00      00
CC          HH      HH  KK      KK  PP      PP  RR      RR  00      00
CC          HH      HH  KK    KK    PP      PP  RR      RR  00      00
CC          HH      HH  KK    KK    PP      PP  RR      RR  00      00
CC          HHHHHHHHHH  KKKKKK      PPPPPPPP    RRRRRRRR    00      00
CC          HHHHHHHHHH  KKKKKK      PPPPPPPP    RRRRRRRR    00      00
CC          HH      HH  KK    KK    PP          RR  RR      00      00
CC          HH      HH  KK    KK    PP          RR  RR      00      00
CC          HH      HH  KK      KK  PP          RR    RR    00      00      ....
CC          HH      HH  KK      KK  PP          RR    RR    00      00      ....
  CCCCCCCC  HH      HH  KK      KK  PP          RR      RR    000000        ....
  CCCCCCCC  HH      HH  KK      KK  PP          RR      RR    000000        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

CHK
V04

```
;     1          0001  0 MODULE CHKPRO (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                     *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                             *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  F11ACP Structure Level 2
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This routine checks the volume and file protection to see if the
;    38          0038  1 !       user is authorized to perform the intended operation.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       STARLET operating system, including privileged system services
;    43          0043  1 !       and internal exec routines.
;    44          0044  1 !
;    45          0045  1 !--
;    46          0046  1 !
;    47          0047  1 !
;    48          0048  1 ! AUTHOR:       L. Mark Pilant,         CREATION DATE: 31-Mar-1983  10:10
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V03-021 LMP0259         L. Mark Pilant,         25-Jun-1984  11:24
;    53          0053  1 !               Remove the clearing of the matching ACE storage.  It has moved
;    54          0054  1 !               to the READ_ATTRIB routine.
;    55          0055  1 !
;    56          0056  1 !       V03-020 ACG0427         Andrew C. Goldstein,    8-May-1984  11:58
;    57          0057  1 !               Finish security auditing. Restructure the saved audit
```

```
;    58          0058  1 !                 block to save space.
;    59          0059  1 !
;    60          0060  1 !           V03-019 ACG0424         Andrew C. Goldstein,    27-Apr-1984  14:28
;    61          0061  1 !                   Filter out local setting of SYSPRV; go back to LOCAL_ARB
;    62          0062  1 !                   for access rights.
;    63          0063  1 !
;    64          0064  1 !           V03-018 LMP0228         L. Mark Pilant,         10-Apr-1984  9:18
;    65          0065  1 !                   Ignore a corrupted ACL during a protection check.
;    66          0066  1 !
;    67          0067  1 !           V03-017 LMP0221         L. Mark Pilant,         7-Apr-1984  13:23
;    68          0068  1 !                   Change the actual protection check to use the new CHKPRO
;    69          0069  1 !                   interface.
;    70          0070  1 !
;    71          0071  1 !           V03-016 RSH0118         R. Scott Hanna          30-Mar-1984
;    72          0072  1 !                   Enable security alarms and make changes due to the new
;    73          0073  1 !                   auditing argument list. Move the READ_HEADER, FID_TO_SPEC,
;    74          0074  1 !                   and NSA$EVENT_AUDIT calls to the DISPAT module.
;    75          0075  1 !
;    76          0076  1 !           V03-015 ACG0412         Andrew C. Goldstein,    25-Mar-1984  17:43
;    77          0077  1 !                   Make all of global storage based, add access mode arg
;    78          0078  1 !
;    79          0079  1 !           V03-014 LMP0208         L. Mark Pilant,         9-Mar-1984  9:16
;    80          0080  1 !                   Don't include the ACL in the protection check if it is
;    81          0081  1 !                   corrupt.  It is still built, however.
;    82          0082  1 !
;    83          0083  1 !           V03-013 LMP0195         L. Mark Pilant,         27-Feb-1984  14:41
;    84          0084  1 !                   Modify the protection checking routine to get the
;    85          0085  1 !                   classification info from the correct place (FCB or header).
;    86          0086  1 !
;    87          0087  1 !           V03-012 LMP0188         L. Mark Pilant,         4-Feb-1984  11:40
;    88          0088  1 !                   Add support for a classification protection check.
;    89          0089  1 !
;    90          0090  1 !           V03-011 RSH0095         R. Scott Hanna          02-Feb-1984
;    91          0091  1 !                   Temporary disable of security auditing.
;    92          0092  1 !
;    93          0093  1 !           V03-010 CDS0001         Christian D. Saether    19-Dec-1983
;    94          0094  1 !                   Use L_NORM linkage and BIND_COMMON macro.
;    95          0095  1 !
;    96          0096  1 !           V03-009 LMP0164         L. Mark Pilant,         10-Oct-1983  15:40
;    97          0097  1 !                   Un-do a bug introduced by ACG0354.  The problem was that you
;    98          0098  1 !                   always got a 5 (ACL) segment descriptor, rather than what was
;    99          0099  1 !                   actually needed.
;   100          0100  1 !
;   101          0101  1 !           V03-008 LMP0158         L. Mark Pilant,         28-Sep-1983  11:19
;   102          0102  1 !                   Insure block type and size are cleared where needed.  (This
;   103          0103  1 !                   was undone by ACG0354.
;   104          0104  1 !
;   105          0105  1 !           V03-007 ACG0354         Andrew C. Goldstein,    12-Sep-1983  18:30
;   106          0106  1 !                   Add CONTROL access via READALL privilege; add
;   107          0107  1 !                   alternate access mask; add ACL driven audit support.
;   108          0108  1 !
;   109          0109  1 !           V03-006 ACG0354         Andrew C. Goldstein,    24-Aug-1983  20:37
;   110          0110  1 !                   Fix setup of protection mask and privilege mask
;   111          0111  1 !
;   112          0112  1 !           V03-005 LMP0134         L. Mark Pilant,         5-Aug-1983  12:30
;   113          0113  1 !                   Fix a problem that caused the access allowed to be incorrectly
;   114          0114  1 !                   returned.
```

```
;  115     0115  1 !
;  116     0116  1 !       V03-004 RSH0034         R. Scott Hanna          05-Jul-1983
;  117     0117  1 !               Add security auditing support.
;  118     0118  1 !
;  119     0119  1 !       V03-003 LMP0121         L. Mark Pilant,         16-Jun-1983  15:52
;  120     0120  1 !               Correct problems with implied protection.
;  121     0121  1 !
;  122     0122  1 !       V03-002 LMP0110         L. Mark Pilant,         3-May-1983  12:15
;  123     0123  1 !               Add support for returning access allowed, privileges used,
;  124     0124  1 !               and the ACE used to gain access (if any).
;  125     0125  1 !
;  126     0126  1 !       V03-001 LMP0104         L. Mark Pilant,         22-Apr-1983  8:50
;  127     0127  1 !               Correct some problem with the rewrite to use $CHKPRO.
;  128     0128  1 !
;  129     0129  1 !**
;  130     0130  1
;  131     0131  1
;  132     0132  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;  133     0133  1
;  134     0134  1 REQUIRE 'SRC$:FCPDEF.B32';
;  135     1125  1
;  136     1126  1 FORWARD ROUTINE
;  137     1127  1         CHECK_PROT      : L_NORM;
;  138     1128  1
;  139     1129  1 BIND                                    ! File access bits
;  140     1130  1         FILE_ACCESS     = UPLIT BYTE (
;  141     1131  1                         ARM$M_READ,
;  142     1132  1                         ARM$M_READ OR ARM$M_WRITE,
;  143     1133  1                         ARM$M_DELETE,
;  144     1134  1                         ARM$M_WRITE,
;  145     1135  1                         ARM$M_READ,
;  146     1136  1                         ARM$M_CONTROL,
;  147     1137  1                         ARM$M_EXECUTE
;  148     1138  1                         ) : VECTOR [,BYTE];
```

```
;  150     1139  1 GLOBAL ROUTINE CHECK_PROTECT (ACCESS, HEADER, FCB, ACMODE, ALT_ACCESS, REQUIRED) : L_NORM =
;  151     1140  1
;  152     1141  1 !++
;  153     1142  1 !
;  154     1143  1 ! FUNCTIONAL DESCRIPTION:
;  155     1144  1 !
;  156     1145  1 !       This routine calls CHECK_PROT and then, if enabled, collects data
;  157     1146  1 !       for file access auditing.
;  158     1147  1 !
;  159     1148  1 ! CALLING SEQUENCE:
;  160     1149  1 !       CHECK_PROTECT (ARG1, ARG2, ARG3, ARG4, ARG5, ARG6)
;  161     1150  1 !
;  162     1151  1 ! INPUT PARAMETERS:
;  163     1152  1 !       The input parameters are passed unmodified to CHECK_PROT. A description
;  164     1153  1 !       of the parameters may be found there.
;  165     1154  1 !
;  166     1155  1 ! OUTPUT PARAMETERS:
;  167     1156  1 !       NONE
;  168     1157  1 !
;  169     1158  1 ! IMPLICIT OUTPUTS:
;  170     1159  1 !       If auditing is enabled for the requested file access, a partial
;  171     1160  1 !       auditing argument list is built in AUDIT_ARGLIST and the counter
;  172     1161  1 !       AUDIT_COUNT is updated. The DISPAT module contains the code which
;  173     1162  1 !       completes the argument list and calls the auditing routine
;  174     1163  1 !       NSA$EVENT_AUDIT.
;  175     1164  1 !
;  176     1165  1 ! ROUTINE VALUE:
;  177     1166  1 !       NONE
;  178     1167  1 !
;  179     1168  1 ! SIDE EFFECTS:
;  180     1169  1 !       NONE
;  181     1170  1 !
;  182     1171  1 !--
;  183     1172  1
;  184     1173  2 BEGIN
;  185     1174  2
;  186     1175  2 MAP
;  187     1176  2         FCB             : REF BBLOCK;   ! FCB arg
;  188     1177  2
;  189     1178  2 LOCAL
;  190     1179  2         STATUS,                         ! Status returned from CHECK_PROT
;  191     1180  2         LOC_ALT_ACCESS,                 ! Local copy of ALT_ACCESS arg
;  192     1181  2         LOC_REQUIRED,                   ! Local copy of REQUIRED arg
;  193     1182  2         LOC_ACCESS,                     ! local version of access mask
;  194     1183  2         ACL_FLAGS       : BITVECTOR [8], ! Audit request flags from ACL
;  195     1184  2         AUDIT_FLAGS     : BITVECTOR [8], ! Flags for audit call
;  196     1185  2         JOURN_MASK,                     ! Accumulated mask of eligible journal events
;  197     1186  2         ALARM_MASK,                     ! Accumulated mask of eligible alarm events
;  198     1187  2         PCB             : REF $BBLOCK,  ! Address of PCB
;  199     1188  2         ARGLIST         : REF $BBLOCK,  ! Argument list pointer
;  200     1189  2         J;
;  201     1190  2
;  202     1191  2 EXTERNAL
;  203     1192  2         NSA$GR_JOURNVEC : $BBLOCK ADDRESSING_MODE (GENERAL),
;  204     1193  2                                         ! Journaling enable bit vector
;  205     1194  2         NSA$GR_ALARMVEC : $BBLOCK ADDRESSING_MODE (GENERAL),
;  206     1195  2                                         ! Alarm enable bit vector
```

```
;  207          1196  2          SCH$GL_CURPCB   : LONG ADDRESSING_MODE (GENERAL);
;  208          1197  2                                          ! Current PCB address
;  209          1198  2
;  210          1199  2 BIND_COMMON;
;  211          1200  2
;  212          1201  2 ! Default the optional arguments to zero.
;  213          1202  2 !
;  214          1203  2
;  215          1204  2 LOC_ALT_ACCESS = 0;
;  216          1205  2 LOC_REQUIRED = 0;
;  217          1206  2 IF ACTUALCOUNT GEQU 6
;  218          1207  2 THEN
;  219          1208  3     BEGIN
;  220          1209  3     LOC_ALT_ACCESS = .ALT_ACCESS;
;  221          1210  3     LOC_REQUIRED = .REQUIRED;
;  222          1211  2     END;
;  223          1212  2
;  224          1213  2 ! Perform protection check
;  225          1214  2
;  226          1215  2 STATUS = CHECK_PROT (.ACCESS, .FCB, .ACMODE, .LOC_ALT_ACCESS, .LOC_REQUIRED, ACL_FLAGS);
;  227          1216  2
;  228          1217  2 ! If the FCB is zero, this is a volume check and no
;  229          1218  2 ! security auditing is performed.
;  230          1219  2
;  231          1220  2 IF .FCB NEQ 0
;  232          1221  2 THEN
;  233          1222  3     BEGIN
;  234          1223  3
;  235          1224  3     LOC_ACCESS = .FILE_ACCESS[.ACCESS];
;  236          1225  4     IF (
;  237          1226  4         IF .STATUS
;  238          1227  4         THEN .STATUS NEQ SS$_NOTALLPRIV
;  239          1228  4         ELSE .REQUIRED
;  240          1229  4         )
;  241          1230  3     THEN LOC_ACCESS = .LOC_ACCESS OR .LOC_ALT_ACCESS;
;  242          1231  3
;  243          1232  3 ! Determine if journaling or alarms are enabled for the
;  244          1233  3 ! specified file access.
;  245          1234  3 !
;  246          1235  3
;  247          1236  3     AUDIT_FLAGS = 0;
;  248          1237  3     JOURN_MASK = .NSA$GR_JOURNVEC[NSA$L_EVT_FAILURE];
;  249          1238  3     ALARM_MASK = .NSA$GR_ALARMVEC[NSA$L_EVT_FAILURE];
;  250          1239  3     IF .STATUS
;  251          1240  3     THEN
;  252          1241  4         BEGIN
;  253          1242  4         JOURN_MASK = .NSA$GR_JOURNVEC[NSA$L_EVT_SUCCESS];
;  254          1243  4         ALARM_MASK = .NSA$GR_ALARMVEC[NSA$L_EVT_SUCCESS];
;  255          1244  4         INCR J FROM 0 TO $BITPOSITION (CHP$V_READALL) DO
;  256          1245  5             BEGIN
;  257          1246  5             IF (.PRIVS_USED AND 1^.J) NEQU 0
;  258          1247  5             THEN
;  259          1248  6                 BEGIN
;  260          1249  6                 JOURN_MASK = .JOURN_MASK OR .VECTOR [NSA$GR_JOURNVEC[NSA$L_EVT_SYSPRV], .J];
;  261          1250  6                 ALARM_MASK = .ALARM_MASK OR .VECTOR [NSA$GR_ALARMVEC[NSA$L_EVT_SYSPRV], .J];
;  262          1251  5                 END;
;  263          1252  5             END
```

```
;  264     1253  3          END;
;  265     1254
;  266     1255  3      PCB = .SCH$GL_CURPCB;
;  267     1256  3      IF .PCB[PCB$V_SECAUDIT]
;  268     1257  3      THEN
;  269     1258  3          AUDIT_FLAGS[$BITPOSITION (NSA$V_ARG_FLAG_MANDY)] = 1;
;  270     1259  3
;  271     1260  3      IF ((.JOURN_MASK AND .LOC_ACCESS) NEQU 0) OR
;  272     1261  4         (.NSA$GR_JOURNVEC[NSA$V_EVT_ACL] AND .ACL_FLAGS[0])
;  273     1262  3      THEN
;  274     1263  3          AUDIT_FLAGS[$BITPOSITION (NSA$V_ARG_FLAG_JOURN)] = 1;
;  275     1264  3
;  276     1265  3      IF ((.ALARM_MASK AND .LOC_ACCESS) NEQU 0) OR
;  277     1266  4         (.NSA$GR_ALARMVEC[NSA$V_EVT_ACL] AND .ACL_FLAGS[1])
;  278     1267  3      THEN
;  279     1268  3          AUDIT_FLAGS[$BITPOSITION (NSA$V_ARG_FLAG_ALARM)] = 1;
;  280     1269  3
;  281     1270  3  ! If journaling, alarms, or mandatory auditing are enabled, find an
;  282     1271  3  ! available audit block and fill it in. Acquiring the file name and
;  283     1272  3  ! sending the audit record is done later.
;  284     1273  3  !
;  285     1274  3
;  286     1275  3      IF .AUDIT_FLAGS NEQ 0
;  287     1276  3      THEN
;  288     1277  4          BEGIN
;  289     1278  4          IF
;  290     1279  5              BEGIN
;  291     1280  5              ARGLIST = AUDIT_ARGLIST;
;  292     1281  5              DECR J FROM MAX_AUDIT_COUNT TO 1
;  293     1282  5              DO
;  294     1283  6                  BEGIN
;  295     1284  6                  IF .ARGLIST[AUDIT_TYPE] EQL 0
;  296     1285  6                  THEN EXITLOOP 0;
;  297     1286  6                  ARGLIST = .ARGLIST + AUDIT_LENGTH;
;  298     1287  6                  END
;  299     1288  5              END
;  300     1289  4          THEN BUG_CHECK (NOBUFPCKT, 'Out of audit list entries');
;  301     1290  4
;  302     1291  4          AUDIT_COUNT = .AUDIT_COUNT + 1;
;  303     1292  4          ARGLIST[AUDIT_TYPE] = .AUDIT_FLAGS;
;  304     1293  4          ARGLIST[AUDIT_SUCCESS] = .STATUS;
;  305     1294  4          ARGLIST[AUDIT_ACCESS] = .LOC_ACCESS;
;  306     1295  4          ARGLIST[AUDIT_PRIVS] = .PRIVS_USED;
;  307     1296  4          CH$MOVE (FCB$S_FID, FCB[FCB$W_FID], ARGLIST[AUDIT_FID]);
;  308     1297  3          END;
;  309     1298  2      END;
;  310     1299  2
;  311     1300  2  IF NOT .STATUS THEN ERR_EXIT (.STATUS) ELSE RETURN .STATUS;
;  312     1301  1  END;


                                                        .TITLE  CHKPRO
                                                        .IDENT  \V04-000\

                                                        .PSECT  $CODE$,NOWRT,2

                        04  10  01  02  08  03  01  00000 P.AAA:  .BYTE   1, 3, 8, 2, 1, 16, 4                    ;
```

```
                                        FILE_ACCESS=        P.AAA
                                                .EXTRN  NSA$GR_JOURNVEC
                                                .EXTRN  NSA$GR_ALARMVEC
                                                .EXTRN  SCH$GL_CURPCB, BUG$_NOBUFPCKT

                                01FC 00000      .ENTRY  CHECK_PROTECT, Save R2,R3,R4,R5,R6,R7,R8    ; 1139
                58 00000000G    00   9E 00002  MOVAB   NSA$GR_ALARMVEC+8, R8                       ;
                57 00000000G    00   9E 00009  MOVAB   NSA$GR_JOURNVEC+8, R7                      ;
                5E                04   C2 00010  SUBL2   #4, SP                                   ;
                                  52   D4 00013  CLRL    LOC_ALT_ACCESS                           ; 1204
                                  50   D4 00015  CLRL    LOC_REQUIRED                             ; 1205
                06                6C   91 00017  CMPB    (AP), #6                                 ; 1206
                                  08   1F 0001A  BLSSU   1$                                       ;
                52          14    AC   D0 0001C  MOVL    ALT_ACCESS, LOC_ALT_ACCESS               ; 1209
                50          18    AC   D0 00020  MOVL    REQUIRED, LOC_REQUIRED                   ; 1210
                          4001    8F   BB 00024  1$:     PUSHR   #^M<R0,SP>                       ; 1215
                                  52   DD 00028  PUSHL   LOC_ALT_ACCESS                           ;
                7E          0C    AC   7D 0002A  MOVQ    FCB, -(SP)                               ;
                            04    AC   DD 0002E  PUSHL   ACCESS                                   ;
          0000V CF                  06   FB 00031  CALLS   #6, CHECK_PROT                       ;
                56                  50   D0 00036  MOVL    R0, STATUS                           ;
                            0C    AC   D5 00039  TSTL    FCB                                      ; 1220
                                  7B   13 0003C  BEQL    13$                                      ;
                50          B8    AF   9E 0003E  MOVAB   FILE_ACCESS, R0                          ; 1224
                55          04 BC40   9A 00042  MOVZBL  @ACCESS[R0], LOC_ACCESS                  ;
                0B                  56   E9 00047  BLBC    STATUS, 2$                           ; 1226
          00000681  8F              56   D1 0004A  CMPL    STATUS, #1665                        ; 1227
                                    09   13 00051  BEQL    4$                                   ;
                                    04   11 00053  BRB     3$                                   ;
                03          18    AC   E9 00055  2$:     BLBC    REQUIRED, 4$                     ; 1228
                55                  52   C8 00059  3$:     BISL2   LOC_ALT_ACCESS, LOC_ACCESS       ; 1230
                                    54   94 0005C  4$:     CLRB    AUDIT_FLAGS                      ; 1236
                53                  67   D0 0005E  MOVL    NSA$GR_JOURNVEC+8, JOURN_MASK        ; 1237
                52                  68   D0 00061  MOVL    NSA$GR_ALARMVEC+8, ALARM_MASK        ; 1238
                22                  56   E9 00064  BLBC    STATUS, 7$                           ; 1239
                53          04    A7   D0 00067  MOVL    NSA$GR_JOURNVEC+12, JOURN_MASK           ; 1242
                52          04    A8   D0 0006B  MOVL    NSA$GR_ALARMVEC+12, ALARM_MASK           ; 1243
                                  50   D4 0006F  CLRL    J                                        ; 1244
          51                      01   78 00071  5$:     ASHL    J, #1, R1                        ; 1246
                51          C4    AA   D3 00075  BITL    -60(BASE), R1                            ;
                                  0A   13 00079  BEQL    6$                                       ;
                53          08 A740   C8 0007B  BISL2   NSA$GR_JOURNVEC+16[J], JOURN_MASK        ; 1249
                52          08 AB40   C8 00080  BISL2   NSA$GR_ALARMVEC+16[J], ALARM_MASK        ; 1250
          E8                    50   F3 00085  6$:     AOBLEQ  #5, J, 5$                        ; 1244
                50 00000000G    00   D0 00089  7$:     MOVL    SCH$GL_CURPCB, PCB               ; 1255
          03        27          A0   03   E1 00090  BBC     #3, 39(PCB), 8$                      ; 1256
                54                  04   88 00095  BISB2   #4, AUDIT_FLAGS                      ; 1258
                55                  53   D3 00098  8$:     BITL    JOURN_MASK, LOC_ACCESS           ; 1260
                                    07   12 0009B  BNEQ    9$                                   ;
                06          F8    A7   E9 0009D  BLBC    NSA$GR_JOURNVEC, 10$                     ; 1261
                03                  6E   E9 000A1  BLBC    ACL_FLAGS, 10$                       ;
                54                  02   88 000A4  9$:     BISB2   #2, AUDIT_FLAGS                  ; 1263
                55                  52   D3 000A7  10$:    BITL    ALARM_MASK, LOC_ACCESS           ; 1265
                                    08   12 000AA  BNEQ    11$                                  ;
                07          F8    A8   E9 000AC  BLBC    NSA$GR_ALARMVEC, 12$                     ; 1266
          03                    6E   01   E1 000B0  BBC     #1, ACL_FLAGS, 12$                   ;
```

```
                                54              01  88 000B4 11$:    BISB2   #1, AUDIT_FLAGS                          ; 1268
                                                54  95 000B7 12$:    TSTB    AUDIT_FLAGS                             ; 1275
                                                36  13 000B9 13$:    BEQL    16$                                     ;
                                50      0924    CA  9E 000BB        MOVAB   2340(BASE), ARGLIST                     ; 1280
                                51              04  D0 000C0        MOVL    #4, J                                   ; 1281
                                                60  95 000C3 14$:    TSTB    (ARGLIST)                               ; 1284
                                                0A  13 000C5        BEQL    15$                                     ;
                                50              10  C0 000C7        ADDL2   #16, ARGLIST                            ; 1286
                                F6              51  F5 000CA        SOBGTR  J, 14$                                  ; 1281
                                                  FEFF 000CD        BUGW                                            ; 1289
                                                 0000* 000CF        .WORD   <BUG$_NOBUFPCKT!4>                      ;
                                        02E4    CA  D6 000D1 15$:    INCL    740(BASE)                               ; 1291
                                60              54  90 000D5        MOVB    AUDIT_FLAGS, (ARGLIST)                  ; 1292
    01  A0              01      00              56  F0 000D8        INSV    STATUS, #0, #1, 1(ARGLIST)              ; 1293
                        08  A0                  55  D0 000DE        MOVL    LOC_ACCESS, 8(ARGLIST)                  ; 1294
                        0C  A0          C4      AA  D0 000E2        MOVL    -60(BASE), 12(ARGLIST)                  ; 1295
                                51      0C      AC  D0 000E7        MOVL    FCB, R1                                 ; 1296
                02  A0          24  A1          06  28 000EB        MOVC3   #6, 36(R1), 2(ARGLIST)                  ;
                                03              56  E8 000F1 16$:    BLBS    STATUS, 17$                             ; 1300
                                                56  BF 000F4        CHMU    STATUS                                  ;
                                                    04 000F6        RET                                             ;
                                50              56  D0 000F7 17$:    MOVL    STATUS, R0                              ;
                                                    04 000FA        RET                                             ; 1301

; Routine Size:  251 bytes,    Routine Base:  $CODE$ + 0007
```

```
;  314     1302  1 ROUTINE CHECK_PROT (ACCESS, FCB, ACMODE, ALT_ACCESS, REQUIRED, AUDIT_FLAGS)
;  315     1303  1         : L_NORM =
;  316     1304  1
;  317     1305  1 !++
;  318     1306  1 !
;  319     1307  1 ! FUNCTIONAL DESCRIPTION:
;  320     1308  1 !
;  321     1309  1 !       This routine checks the volume and file protection to see if the
;  322     1310  1 !       user is authorized to perform the intended operation.
;  323     1311  1 !
;  324     1312  1 ! CALLING SEQUENCE:
;  325     1313  1 !       CHECK_PROTECTION (ARG1, ARG2, ARG3, ARG4, ARG5, ARG6)
;  326     1314  1 !
;  327     1315  1 ! INPUT PARAMETERS:
;  328     1316  1 !       ARG1: access mode
;  329     1317  1 !               READ_ACCESS     = 0
;  330     1318  1 !               WRITE_ACCESS    = 1
;  331     1319  1 !               DELETE_ACCESS   = 2
;  332     1320  1 !               CREATE_ACCESS   = 3
;  333     1321  1 !               RDATT_ACCESS    = 4
;  334     1322  1 !               WRATT_ACCESS    = 5
;  335     1323  1 !               EXEC_ACCESS     = 6
;  336     1324  1 !       ARG2: address of FCB or 0
;  337     1325  1 !       ARG3: access mode of the accessor
;  338     1326  1 !       ARG4: alternate access mask to test for
;  339     1327  1 !       ARG5: 1 if alternate access if required
;  340     1328  1 !
;  341     1329  1 ! IMPLICIT INPUTS:
;  342     1330  1 !       CURRENT_UCB: address of device UCB
;  343     1331  1 !       IO_PACKET: I/O packet of this request
;  344     1332  1 !
;  345     1333  1 ! OUTPUT PARAMETERS:
;  346     1334  1 !       ARG6: address in which to store audit enable flags
;  347     1335  1 !             bit 0 = enable audit
;  348     1336  1 !             bit 1 = enable alarm
;  349     1337  1 !
;  350     1338  1 ! IMPLICIT OUTPUTS:
;  351     1339  1 !       NONE
;  352     1340  1 !
;  353     1341  1 ! ROUTINE VALUE:
;  354     1342  1 !       NONE
;  355     1343  1 !
;  356     1344  1 ! SIDE EFFECTS:
;  357     1345  1 !       NONE
;  358     1346  1 !
;  359     1347  1 !--
;  360     1348  1
;  361     1349  2 BEGIN
;  362     1350  2
;  363     1351  2 MAP
;  364     1352  2         FCB             : REF BBLOCK,   ! FCB arg
;  365     1353  2         AUDIT_FLAGS     : REF BITVECTOR; ! audit and alarm flags
;  366     1354  2
;  367     1355  2 LINKAGE
;  368     1356  2         L_CHKPRO_INT    = JSB (REGISTER = 0, REGISTER = 1,
;  369     1357  2                                REGISTER = 2, REGISTER = 3);
;  370     1358  2
```

```
;  371     1359  2 LABEL
;  372     1360  2         CHECK_BLOCK;                    ! body of a single check attempt
;  373     1361  2
;  374     1362  2 LOCAL
;  375     1363  2         STATUS,                         ! Local routine exit status
;  376     1364  2         FILE_ACCESS_BITS: BBLOCK [1],   ! Actual access mask to file
;  377     1365  2         PROTECTION,                     ! Protection code of file
;  378     1366  2         OWNER_UIC,                      ! File owner UIC
;  379     1367  2         SEG_NUMBER,                     ! Segment number of file header
;  380     1368  2         AUDIT_BUFFER,                   ! Audit name string buffer
;  381     1369  2         ALARM_BUFFER,                   ! Alarm name string buffer
;  382     1370  2         CHPCTL          : BBLOCK [CHPCTL$C_LENGTH],     ! CHKPRO control block
;  383     1371  2         CHPRET          : BBLOCK [CHPRET$C_LENGTH],     ! CHKPRO return arg block
;  384     1372  2         ORB             : REF BBLOCK,   ! Object's rights block
;  385     1373  2         LOCAL_ORB       : BBLOCK [ORB$C_LENGTH];        ! Used for BADACL checks
;  386     1374  2
;  387     1375  2 BIND                                    ! Access mode tables
;  388     1376  2                                         ! Write operation on volume
;  389     1377  2         WRITE_OP        = UPLIT (
;  390     1378  2                         ARMSM_WRITE OR ARMSM_DELETE OR ARMSM_CONTROL),
;  391     1379  2
;  392     1380  2                                         ! no READALL privilege for operation
;  393     1381  2         NOREADALL       = UPLIT (
;  394     1382  2                         ARMSM_WRITE OR ARMSM_DELETE),
;  395     1383  2
;  396     1384  2                                         ! Check for zero file segment number
;  397     1385  2         EXT_HEADER      = UPLIT BYTE (
;  398     1386  2                         %B'1100111'
;  399     1387  2                         ) : BITVECTOR,
;  400     1388  2
;  401     1389  2                                         ! Volume access bits
;  402     1390  2         VOL_ACCESS      = UPLIT BYTE (
;  403     1391  2                         ARMSM_READ,
;  404     1392  2                         ARMSM_READ OR ARMSM_WRITE,
;  405     1393  2                         ARMSM_READ OR ARMSM_DELETE,
;  406     1394  2                         ARMSM_READ OR ARMSM_WRITE OR ARMSM_EXECUTE,
;  407     1395  2                         ARMSM_READ,
;  408     1396  2                         ARMSM_READ OR ARMSM_WRITE,
;  409     1397  2                         ARMSM_READ
;  410     1398  2                         ) : VECTOR [,BYTE];
;  411     1399  2
;  412     1400  2
;  413     1401  2 EXTERNAL
;  414     1402  2         EXE$GL_DYNAMIC_FLAGS    : BITVECTOR ADDRESSING_MODE (ABSOLUTE);
;  415     1403  2
;  416     1404  2 EXTERNAL LITERAL
;  417     1405  2         EXE$V_CLASS_PROT;
;  418     1406  2
;  419     1407  2 BIND_COMMON;
;  420     1408  2
;  421     1409  2 EXTERNAL ROUTINE
;  422     1410  2         EXE$CHKPRO_INT  : L_CHKPRO_INT ADDRESSING_MODE (GENERAL);
;  423     1411  2                                         ! General purpose protection checker
;  424     1412  2
;  425     1413  2 ! Initialize storage.
;  426     1414  2
;  427     1415  2 MATCHING_ACE[ACE$B_SIZE] = 0;                   ! Only the size needs initializing
```

```
;  428     1416  2
;  429     1417  2 AUDIT_BUFFER = 0;
;  430     1418  2 ALARM_BUFFER = 0;
;  431     1419  2 PRIVS_USED = 0;
;  432     1420  2
;  433     1421  2 ! Items to return
;  434     1422  2
;  435     1423  2 CHPRET[CHPRET$W_MATCHED_ACELEN] = ATR$S_READACE;
;  436     1424  2 CHPRET[CHPRET$L_MATCHED_ACE] = MATCHING_ACE;
;  437     1425  2 CHPRET[CHPRET$L_MATCHED_ACERET] = 0;
;  438     1426  2 CHPRET[CHPRET$W_AUDITLEN] = 4;
;  439     1427  2 CHPRET[CHPRET$L_AUDIT] = AUDIT_BUFFER;
;  440     1428  2 CHPRET[CHPRET$L_AUDITRET] = 0;
;  441     1429  2 CHPRET[CHPRET$W_ALARMLEN] = 4;
;  442     1430  2 CHPRET[CHPRET$L_ALARM] = ALARM_BUFFER;
;  443     1431  2 CHPRET[CHPRET$L_ALARMRET] = 0;
;  444     1432  2 CHPRET[CHPRET$L_PRIVS_USED] = PRIVS_USED;
;  445     1433  2
;  446     1434  2 ! Derive the composite file access mask from the access type and
;  447     1435  2 ! the alternate access mask.
;  448     1436  2 !
;  449     1437  2
;  450     1438  2 FILE_ACCESS_BITS = .FILE_ACCESS[.ACCESS] OR .ALT_ACCESS;
;  451     1439  2
;  452     1440  2 ! We try the whole operation twice: once with the added alternate access
;  453     1441  2 ! mask, and if that fails, once without.
;  454     1442  2 !
;  455     1443  2
;  456     1444  2 WHILE 1 DO
;  457     1445  3     BEGIN
;  458     1446  4     CHECK_BLOCK: BEGIN                  ! scope of one try
;  459     1447  4
;  460     1448  4 ! If the requested operation is a write operation, check to make
;  461     1449  4 ! sure that the volume is not software write locked.
;  462     1450  4
;  463     1451  4     IF (.WRITE_OP AND .FILE_ACCESS_BITS) NEQ 0
;  464     1452  4     AND .BBLOCK [CURRENT_UCB[UCB$L_DEVCHAR], DEV$V_SWL]
;  465     1453  4     THEN
;  466     1454  5         BEGIN
;  467     1455  5         STATUS = SS$_WRITLCK;
;  468     1456  5         LEAVE CHECK_BLOCK;
;  469     1457  4         END;
;  470     1458  4
;  471     1459  4 ! Get the address of the Object's Rights Block (ORB).
;  472     1460  4
;  473     1461  4     ORB = .CURRENT_UCB[UCB$L_ORB];
;  474     1462  4
;  475     1463  4 ! Now check the volume protection to make sure that the requested operation
;  476     1464  4 ! is allowed.  If the attempted access is denied, return with the error.
;  477     1465  4
;  478     1466  4     CHPCTL[CHPCTL$L_ACCESS] = .VOL_ACCESS[.ACCESS];
;  479     1467  4     IF .FILE_ACCESS_BITS[ARM$V_WRITE]
;  480     1468  4     OR .FILE_ACCESS_BITS[ARM$V_CONTROL]
;  481     1469  4     THEN BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_WRITE] = 1;
;  482     1470  4     IF .FILE_ACCESS_BITS[ARM$V_DELETE]
;  483     1471  4     THEN BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_DELETE] = 1;
;  484     1472  4     CHPCTL[CHPCTL$B_MODE] = 0;
```

```
;  485          1473  4      CHPCTL[CHPCTL$L_FLAGS] = CHP$M_READ;
;  486          1474  4      IF (.WRITE_OP AND .FILE_ACCESS_BITS) NEQ 0
;  487          1475  4      THEN BBLOCK [CHPCTL[CHPCTL$L_FLAGS], CHP$V_WRITE] = 1;
;  488          1476  4      IF (.NOREADALL AND .FILE_ACCESS_BITS) EQL 0
;  489          1477  4      THEN BBLOCK [CHPCTL[CHPCTL$L_FLAGS], CHP$V_USEREADALL] = 1;
;  490          1478  4
;  491          1479  4      STATUS = EXE$CHKPRO_INT (LOCAL_ARB, .ORB, CHPCTL, 0);
;  492          1480  4      IF NOT .STATUS
;  493          1481  4      THEN LEAVE CHECK_BLOCK;
;  494          1482  4
;  495          1483  4  ! If there is no FCB specified, it is a volume access
;  496          1484  4  ! check.  In which case, control may be returned now.
;  497          1485  4
;  498          1486  4      IF .FCB EQL 0
;  499          1487  4      THEN LEAVE CHECK_BLOCK;
;  500          1488  4
;  501          1489  4  ! Get the protection, owner, and segment number for the desired header.
;  502          1490  4  ! Also, get the classification information if doing classification checks.
;  503          1491  4
;  504          1492  4      IF .FCB[FCB$V_BADACL]
;  505          1493  4      THEN
;  506          1494  5          BEGIN
;  507          1495  5          CH$MOVE (ORB$C_LENGTH, FCB[FCB$R_ORB], LOCAL_ORB);
;  508          1496  5          LOCAL_ORB[ORB$V_ACL_QUEUE] = 0;
;  509          1497  5          LOCAL_ORB[ORB$L_ACLFL] = LOCAL_ORB[ORB$L_ACLBL] = 0;
;  510          1498  5          ORB = LOCAL_ORB;
;  511          1499  5          END
;  512          1500  4      ELSE ORB = FCB[FCB$R_ORB];
;  513          1501  4      SEG_NUMBER = .FCB[FCB$W_SEGN];
;  514          1502  4
;  515          1503  4  ! Next, if the operation is on an extension header, make sure that only the
;  516          1504  4  ! system is allowed access for most operations.
;  517          1505  4
;  518          1506  4      IF .EXT_HEADER[.ACCESS]
;  519          1507  4      THEN
;  520          1508  5          BEGIN
;  521          1509  5          IF .SEG_NUMBER GTR 0 AND NOT .CLEANUP_FLAGS[CLF_SYSPRV]
;  522          1510  5          THEN
;  523          1511  6              BEGIN
;  524          1512  6              STATUS = SS$_NOPRIV;
;  525          1513  6              LEAVE CHECK_BLOCK;
;  526          1514  5              END;
;  527          1515  4          END;
;  528          1516  4
;  529          1517  4  ! Now check the access requested to determine if access is to be granted or
;  530          1518  4  ! denied.
;  531          1519  4
;  532          1520  4      CHPCTL[CHPCTL$L_ACCESS] = .FILE_ACCESS_BITS;
;  533          1521  4      CHPCTL[CHPCTL$B_MODE] = .ACMODE;
;  534          1522  4
;  535          1523  4      STATUS = EXE$CHKPRO_INT (LOCAL_ARB, .ORB, CHPCTL, CHPRET);
;  536          1524  4
;  537          1525  4  ! Certain operations may be permitted by more than one access type.
;  538          1526  4  ! Read implies execute, and control implies read attributes. The
;  539          1527  4  ! protection check needs to be retried in these cases.
;  540          1528  4  !
;  541          1529  4
```

```
;  542          1530  4      IF NOT .STATUS
;  543          1531  4      THEN
;  544          1532  5          BEGIN
;  545          1533  5          IF .ACCESS EQL EXEC_ACCESS
;  546          1534  5          THEN
;  547          1535  6              BEGIN
;  548          1536  6              BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_EXECUTE] = 0;
;  549          1537  6              BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_READ] = 1;
;  550          1538  6              AUDIT_BUFFER = 0;
;  551          1539  6              ALARM_BUFFER = 0;
;  552          1540  6              PRIVS_USED = 0;
;  553          1541  6              STATUS = EXE$CHKPRO_INT (LOCAL_ARB, .ORB, CHPCTL, CHPRET);
;  554          1542  6              END
;  555          1543  5          ELSE IF .ACCESS EQL RDATT_ACCESS
;  556          1544  5          THEN
;  557          1545  6              BEGIN
;  558          1546  6              BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_READ] = 0;
;  559          1547  6              BBLOCK [CHPCTL[CHPCTL$L_ACCESS], ARM$V_CONTROL] = 1;
;  560          1548  6              AUDIT_BUFFER = 0;
;  561          1549  6              ALARM_BUFFER = 0;
;  562          1550  6              PRIVS_USED = 0;
;  563          1551  6              STATUS = EXE$CHKPRO_INT (LOCAL_ARB, .ORB, CHPCTL, CHPRET);
;  564          1552  5              END;
;  565          1553  4          END;
;  566          1554  4
;  567          1555  4  ! If we just tried a protection check with alternate access and it
;  568          1556  4  ! failed, retry it with just the normal access. Otherwise, we are
;  569          1557  4  ! done.
;  570          1558  4  !
;  571          1559  4
;  572          1560  3      END;                                        ! end of block CHECK_BLOCK
;  573          1561  3
;  574          1562  3      IF .STATUS
;  575          1563  3      OR .REQUIRED
;  576          1564  3      OR .FILE_ACCESS_BITS EQL .FILE_ACCESS[.ACCESS]
;  577          1565  3      THEN EXITLOOP;
;  578          1566  3
;  579          1567  3      FILE_ACCESS_BITS = .FILE_ACCESS[.ACCESS];
;  580          1568  2      END;                                        ! end of retry loop
;  581          1569  2
;  582          1570  2  ! Return audit and alarm status.
;  583          1571  2  !
;  584          1572  2
;  585          1573  2  .AUDIT_FLAGS = 0;
;  586          1574  2  IF .AUDIT_BUFFER NEQ 0
;  587          1575  2  THEN AUDIT_FLAGS[0] = 1;
;  588          1576  2  IF .ALARM_BUFFER NEQ 0
;  589          1577  2  THEN AUDIT_FLAGS[1] = 1;
;  590          1578  2
;  591          1579  2  ! Check if the alternate access check failed. If so, return alternate
;  592          1580  2  ! success status.
;  593          1581  2  !
;  594          1582  2
;  595          1583  2  IF .STATUS
;  596          1584  2  AND .ALT_ACCESS NEQ 0
;  597          1585  2  AND .FILE_ACCESS_BITS EQL .FILE_ACCESS[.ACCESS]
;  598          1586  2  THEN STATUS = SS$_NOTALLPRIV;
```

```
;  599          1587  2
;  600          1588  2 ! Postprocess setting of the SYSPRV priv used bit. We set SYSPRV in the
;  601          1589  2 ! local privilege mask under various circumstances (e.g., volume ownership),
;  602          1590  2 ! but only want to log it if the caller really had it.
;  603          1591  2 !
;  604          1592  2
;  605          1593  2 IF .PRIVS_USED[CHP$V_SYSPRV]
;  606          1594  2 AND .CLEANUP_FLAGS[CLF_VOLOWNER]
;  607          1595  2 THEN
;  608          1596  3     BEGIN
;  609          1597  3     PRIVS_USED[CHP$V_SYSPRV] = 0;
;  610          1598  3     IF .CLEANUP_FLAGS[CLF_GRPOWNER]
;  611          1599  3     THEN PRIVS_USED[CHP$V_GRPPRV] = 1;
;  612          1600  2     END;
;  613          1601  2
;  614          1602  2 RETURN .STATUS
;  615          1603  2
;  616          1604  1 END;                                   ! End of routine CHECK_PROTECT


                                        00102                .BLKB   2
                                0000001A 00104 P.AAB:        .LONG   26                                          ;
                                0000000A 00108 P.AAC:        .LONG   10                                          ;
                                      67 0010C P.AAD:        .BYTE   103                                         ;
                   01  03  01  07  09  03  01 0010D P.AAE:    .BYTE   1, 3, 9, 7, 1, 3, 1                         ;

                                                 WRITE_OP=           P.AAB
                                                 NOREADALL=          P.AAC
                                                 EXT_HEADER=         P.AAD
                                                 VOL_ACCESS=         P.AAE
                                                        .EXTRN  EXE$GL_DYNAMIC_FLAGS
                                                        .EXTRN  EXE$V_CLASS_PROT
                                                        .EXTRN  EXE$CHKPRO_INT

                                     0BFC 00000 CHECK_PROT:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R11            ; 1302
                         5E      FF60   CE  9E 00002            MOVAB   -160(SP), SP
                   08    AE        C4   AA  9E 00007            MOVAB   -60(BASE), 8(SP)                    ; 1405
                   04    AE      0284   CA  9E 0000C            MOVAB   644(BASE), 4(SP)
                               02E8   CA  94 00012            CLRB    744(BASE)                           ; 1415
                                 0C   AE  7C 00016            CLRQ    AUDIT_BUFFER                        ; 1417
                                 08   BE  D4 00019            CLRL    @8(SP)                              ; 1419
                   E4    AD        FF   8F  9B 0001C            MOVZBW  #255, CHPRET+24                     ; 1423
                   E8    AD      02E8   CA  9E 00021            MOVAB   744(BASE), CHPRET+28                ; 1424
                                 EC   AD  D4 00027            CLRL    CHPRET+32                           ; 1425
                   6C    AE           04  B0 0002A            MOVW    #4, CHPRET                          ; 1426
                   70    AE        0C   AE  9E 0002E            MOVAB   AUDIT_BUFFER, CHPRET+4              ; 1427
                                 74   AE  D4 00033            CLRL    CHPRET+8                            ; 1428
                   78    AE           04  B0 00036            MOVW    #4, CHPRET+12                       ; 1429
                   7C    AE        10   AE  9E 0003A            MOVAB   ALARM_BUFFER, CHPRET+16             ; 1430
                                 E0   AD  D4 0003F            CLRL    CHPRET+20                           ; 1431
                   F0    AD        08   AE  D0 00042            MOVL    8(SP), CHPRET+36                    ; 1432
                         50      FEA1   CF  9E 00047            MOVAB   FILE_ACCESS, R0                     ; 1438
                58       04 BC40   10   AC  89 0004C            BISB3   ALT_ACCESS, @ACCESS[R0], FILE_ACCESS_BITS
                               57   04   AC  D0 00053            MOVL    ACCESS, R7                          ; 1466
                                         51  D4 00057 1$:       CLRL    R1                                  ; 1451
```

```
                58      94    AF   93 00059          BITB    WRITE_OP, FILE_ACCESS_BITS                      ;
                              13   13 0005D          BEQL    3$                                              ;
                              51   D6 0005F          INCL    R1                                              ;
                50      94    AA   D0 00061          MOVL    -108(BASE), R0                                  ; 1452
          08    3B    A0          01 E1 00065          BBC     #1, 59(R0), 3$                                  ;
                59    025C  8F    3C 0006A          MOVZWL  #604, STATUS                                    ; 1455
                      00F4  31    0006F 2$:         BRW     15$                                             ; 1456
                50      94    AA   D0 00072 3$:       MOVL    -108(BASE), R0                                  ; 1461
                5B      1C    A0   D0 00076          MOVL    28(R0), ORB                                     ;
                F4    AD    FF7A  CF47   9A 0007A          MOVZBL  VOL_ACCESS[R7], CHPCTL                  ; 1466
          04          58          01 E0 00081          BBS     #1, FILE_ACCESS_BITS, 4$                        ; 1467
          04          58          04 E1 00085          BBC     #4, FILE_ACCESS_BITS, 5$                        ; 1468
                F4    AD          02 88 00089 4$:       BISB2   #2, CHPCTL                                      ; 1469
          04          58          03 E1 0008D 5$:       BBC     #3, FILE_ACCESS_BITS, 6$                        ; 1470
                F4    AD          08 88 00091          BISB2   #8, CHPCTL                                      ; 1471
                        FC    AD   94 00095 6$:       CLRB    CHPCTL+8                                        ; 1472
                F8    AD          01 D0 00098          MOVL    #1, CHPCTL+4                                    ; 1473
                04                51 E9 0009C          BLBC    R1, 7$                                          ; 1474
                F8    AD          02 88 0009F          BISB2   #2, CHPCTL+4                                    ; 1475
                58      FF4D  CF    93 000A3 7$:       BITB    NOREADALL, FILE_ACCESS_BITS                     ; 1476
                              04   12 000A8          BNEQ    8$                                              ;
                F8    AD          04 88 000AA          BISB2   #4, CHPCTL+4                                    ; 1477
                52      F4    AD   9E 000AE 8$:       MOVAB   CHPCTL, R2                                      ; 1479
                              53   D4 000B2          CLRL    R3                                              ;
                51              5B   D0 000B4          MOVL    ORB, R1                                         ;
                50        04    AE   D0 000B7          MOVL    4(SP), R0                                       ;
                  00000000G   00   16 000BB          JSB     EXE$CHKPRO_INT                                  ;
                59              50   D0 000C1          MOVL    R0, STATUS                                      ;
                03              59   E8 000C4          BLBS    STATUS, 9$                                      ; 1480
                              009F  31   000C7          BRW     16$                                             ;
                56        08    AC   D0 000CA 9$:       MOVL    FCB, R6                                         ; 1486
                              9F   13 000CE          BEQL    2$                                              ;
                        22    A6   95 000D0          TSTB    34(R6)                                          ; 1492
                              15   18 000D3          BGEQ    10$                                             ;
          14    AE      58    A6      005B  8F    28 000D5          MOVC3   #88, 88(R6), LOCAL_ORB                  ; 1495
                1F    AE          02 8A 000DD          BICB2   #2, LOCAL_ORB+11                                ; 1496
                        3C    AE   7C 000E1          CLRQ    LOCAL_ORB+40                                    ; 1497
                5B      14    AE   9E 000E4          MOVAB   LOCAL_ORB, ORB                                  ; 1498
                              04   11 000E8          BRB     11$                                             ; 1492
                5B      58    A6   9E 000EA 10$:      MOVAB   88(R6), ORB                                     ; 1500
                6E      2A    A6   3C 000EE 11$:      MOVZWL  42(R6), SEG_NUMBER                              ; 1501
          0D      FF00  CF          04 AC E1 000F2          BBC     ACCESS, EXT_HEADER, 12$                         ; 1506
                              6E   D5 000F9          TSTL    SEG_NUMBER                                      ; 1509
                              09   15 000FB          BLEQ    12$                                             ;
                05      01    AA   E8 000FD          BLBS    1(BASE), 12$                                    ;
                59              24   D0 00101          MOVL    #36, STATUS                                     ; 1512
                              60   11 00104          BRB     15$                                             ; 1513
                F4    AD          58   9A 00106 12$:      MOVZBL  FILE_ACCESS_BITS, CHPCTL                        ; 1520
                FC    AD    0C    AC   90 0010A          MOVB    ACMODE, CHPCTL+8                                ; 1521
                53      6C    AE   9E 0010F          MOVAB   CHPRET, R3                                      ; 1523
                52      F4    AD   9E 00113          MOVAB   CHPCTL, R2                                      ;
                51              5B   D0 00117          MOVL    ORB, R1                                         ;
                50        04    AE   D0 0011A          MOVL    4(SP), R0                                       ;
                  00000000G   00   16 0011E          JSB     EXE$CHKPRO_INT                                  ;
                59              50   D0 00124          MOVL    R0, STATUS                                      ;
                5C              59   E8 00127          BLBS    STATUS, 17$                                     ; 1530
                06        04    AC   D1 0012A          CMPL    ACCESS, #6                                      ; 1533
```

```
                               0A 12 0012E          BNEQ    13$                                              ;
        F4  AD                 04 8A 00130          BICB2   #4, CHPCTL                                       ; 1536
        F4  AD                 01 88 00134          BISB2   #1, CHPCTL                                       ; 1537
                               0E 11 00138          BRB     14$                                              ; 1538
            04         04      AC D1 0013A 13$:     CMPL    ACCESS, #4                                       ; 1543
                               26 12 0013E          BNEQ    15$                                              ;
        F4  AD                 01 8A 00140          BICB2   #1, CHPCTL                                       ; 1546
        F4  AD                 10 88 00144          BISB2   #16, CHPCTL                                      ; 1547
                       0C      AE 7C 00148 14$:     CLRQ    AUDIT_BUFFER                                     ; 1548
                       08      BE D4 0014B          CLRL    @8(SP)                                           ; 1550
            53         6C      AE 9E 0014E          MOVAB   CHPRET, R3                                       ; 1551
            52         F4      AD 9E 00152          MOVAB   CHPCTL, R2                                       ;
            51                 5B D0 00156          MOVL    ORB, R1                                          ;
            50         04      AE D0 00159          MOVL    4(SP), R0                                        ;
                00000000G      00 16 0015D          JSB     EXE$CHKPRO_INT                                   ;
            59                 50 D0 00163          MOVL    R0, STATUS                                       ;
            1D                 59 E8 00166 15$:     BLBS    STATUS, 17$                                      ; 1562
            19         14      AC E8 00169 16$:     BLBS    REQUIRED, 17$                                    ; 1563
            50       FD7B      CF 9E 0016D          MOVAB   FILE_ACCESS, R0                                  ; 1564
        04 BC40                58 91 00172          CMPB    FILE_ACCESS_BITS, @ACCESS[R0]                    ;
                               0D 13 00177          BEQL    17$                                              ;
            57         04      AC D0 00179          MOVL    ACCESS, R7                                       ; 1567
            58       FD6A CF47    90 0017D          MOVB    FILE_ACCESS[R7], FILE_ACCESS_BITS                ;
                             FED1 31 00183          BRW     1$                                               ; 1444
                       18      BC D4 00186 17$:     CLRL    @AUDIT_FLAGS                                     ; 1573
                       0C      AE D5 00189          TSTL    AUDIT_BUFFER                                     ; 1574
                               04 13 0018C          BEQL    18$                                              ;
        18  BC                 01 88 0018E          BISB2   #1, @AUDIT_FLAGS                                 ; 1575
                       10      AE D5 00192 18$:     TSTL    ALARM_BUFFER                                     ; 1576
                               04 13 00195          BEQL    19$                                              ;
        18  BC                 02 88 00197          BISB2   #2, @AUDIT_FLAGS                                 ; 1577
            16                 59 E9 0019B 19$:     BLBC    STATUS, 20$                                      ; 1583
                       10      AC D5 0019E          TSTL    ALT_ACCESS                                       ; 1584
                               11 13 001A1          BEQL    20$                                              ;
            50       FD45      CF 9E 001A3          MOVAB   FILE_ACCESS, R0                                  ; 1585
        04 BC40                58 91 001A8          CMPB    FILE_ACCESS_BITS, @ACCESS[R0]                    ;
                               05 12 001AD          BNEQ    20$                                              ;
            59       0681      8F 3C 001AF          MOVZWL  #1665, STATUS                                    ; 1586
            10         08      BE E9 001B4 20$:     BLBC    @8(SP), 21$                                      ; 1593
    0C      6A                 0C E1 001B8          BBC     #12, (BASE), 21$                                 ; 1594
        08  BE                 01 8A 001BC          BICB2   #1, @8(SP)                                       ; 1597
    04      6A                 0D E1 001C0          BBC     #13, (BASE), 21$                                 ; 1598
        08  BE                 10 88 001C4          BISB2   #16, @8(SP)                                      ; 1599
            50                 59 D0 001C8 21$:     MOVL    STATUS, R0                                       ; 1602
                               04 001CB             RET                                                      ; 1604

; Routine Size:  460 bytes,    Routine Base:  $CODE$ + 0114


;  617          1605  1
;  618          1606  1 END
;  619          1607  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                         736  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                                 Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619        74         0      1000          00:02.0




;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:CHKPRO/OBJ=OBJ$:CHKPRO MSRC$:CHKPRO/UPDATE=(ENH$:CHKPRO)

; Size:         711 code + 25 data bytes
; Run Time:         00:35.7
; Elapsed Time:     01:04.7
; Lines/CPU Min:     2704
; Lexemes/CPU-Min: 46579
; Memory Used:   297 pages
; Compilation Complete
```

CPYNAM
LIS

CHKDMO
LIS

CLENUP
LIS

CHKPRO
LIS

CHARGEQ
LIS

COMMON
LIS

CREATE
LIS

BADSCN
LIS

CHKHD2
LIS

CHKSUM
LIS

ALLOCB
LIS